REMARKS

In the above-identified Office Action, the Examiner rejected Claims 28 – 31 under 35 U.S.C. §101 as being directed to non-statutory subject matter. Claims 1 – 5, 8 - 20, 23, 24 and 27 - 34 were rejected under 35 U.S.C. §103(a) as being unpatentable over Gish. Claims 6, 7 21, 22, 25 and 26 were rejected under 35 U.S.C. §103(a) as being unpatentable over Gish in view of Nakajima.

In response to the 101 rejection of Claims 28 – 31, Claims 28 and 29 have been amended to include the limitations of a processor processing instructions to perform the function in the claims. Consequently the claims presently include hardware as well as software and thus are now within patentable subject matter.

Hence, Applicants kindly request withdrawal of the 101 rejection.

Further, Applicants amended the Specification to include the term "readable media." Since the term "readable media" is in the original Specification (see original Claim 28) to refer to recordable media on page 7, lines 24 – 27, no new matter is added to the Application by this amendment.

Independent Claims 1, 12, 28, 29 and 32 have been amended to better claim the invention. Particularly, the claims are amended to replace ~~a server~~ with one of a plurality of servers and ~~a client application~~ with one of a plurality of client applications. Support for the added limitations can be found, among other locations, on page 9, lines 1 – 3.

In addition to the amendment of the independent claims, Applicants have amended dependent Claims 2, 3, 5, 6, 8, 10, 12, 19, 20, 30 and 31 to better claim the invention.

By this amendment, Claims 1 – 34 remain pending in the Application. For the reasons stated more fully below, Applicants submit that the pending claims are allowable over the applied references. Hence, reconsideration, allowance and passage to issue are respectfully requested.

The invention is set forth in claims of varying scopes of which Claim 1 is illustrative.

> 1. A method of executing code in a client-server environment comprising:
>
> ***identifying an input object on a client system, the input object identifying code for executing on one of a plurality of servers***;
>
> ***processing the input object to identify the code for executing on the one of the plurality of servers***;
>
> ***generating, in response to identifying the code for executing on the one of the plurality of servers, code for accessing the code for executing on the one of the plurality of servers***;
>
> ***processing the generated code to determine the one of the plurality of servers on which to execute the code, each code for executing on a server being able to execute on a particular server***;
>
> enabling the determined server to access the code for executing on the one of the plurality of servers;
>
> identifying, based on the accessed code for executing on the one of the plurality of servers, one of a plurality of client applications for allowing the determined server to interact with the client system during processing of the code for executing on the one of the plurality of servers; and
>
> processing the code for executing on the one of the plurality of servers. (Emphasis added.)

The Examiner asserted that except for "identifying an input object on a client system, the input object identifying code for executing on a server," Gish teaches the claimed invention in Figs. 4 and 5, col. 17, lines 34 – 52 and col. 8, lines 1 – 30. But, the Examiner continued, "it is common for a person skilled in the art to see that Figs. 4 and 5 of Gish, the client communicates with the server, and helps in executing the application which is stored on the server." Therefore, the Examiner concluded, "it would have been obvious to a person skilled in the art at the time the invention was made to incorporate the use of inputting object on the client system taught by Gish, which is being transferred to the server to execute the application and being presented to the client." Applicants respectfully disagree.

Firstly, since as admitted by the Examiner, Gish does not teach ***identifying an input object on a client system, the input object identifying code for executing on a server***, Gish then cannot teach, show or suggest ***processing the input object to identify the code for executing on the one of the plurality of servers*** as asserted by the Examiner.

Secondly and as mentioned in the Response to the Previous Office Action, Gish purports to teach an enterprise computing manager in which an application is composed of a client (front end) program which communicates utilizing a network with a server (back end) program. The client and server programs are loosely coupled and exchange information using the network. The client program is composed of a User Interface (UI) and an object-oriented framework (Presentation Engine (PE) framework). The UI exchanges data messages with the framework. The framework is designed to handle two types of messages: (1) messages from the UI, and (2) messages from the server (back end) program via the network. The framework includes a component, the mediator which manages messages coming into and going out of the framework.

In col. 18, lines 7 – 30, Gish discloses:

> FIG. 5 illustrates how a preferred embodiment leverages Java to facilitate the establishment and implementation of server-centric policies. The client and server nodes communicate utilizing the web technologies in an Internet, Intranet or other network environment. The client node 500 contacts the server node 520 via HTTP with a request to execute an application. After authenticating the client 500, the server node 520 selects front 502 and back end 510 components based on the application definition list maintained at the server node 520. The server starts its back end process 510 and sends the front end program 502 to the client node via the Web technologies in an Internet, Intranet or other network environment. The client 500 executes the selected front end 502 locally at the client node 500. The front end (client)

> programs open a TCP/IP connection back to the server to initiate message passing in order to run the applications. The front end program 502 is implemented entirely in Java which facilitates instances of client/server applications which can run concurrently on a set of multi-platform clients. The server 520 is able to send a front end program 502 to any client node 500 which has the Java runtime installed on the computer. Server policies will not involve the clients. The policies will focus on the server's control of its local resources. (Emphasis added.)

According to the above-reproduced paragraph, a client sends a request to a server to execute an application (e.g., the code). The server identifies, based on the application to be executed, a front end component 502 (e.g., a client application) and a back end component 510 that will facilitate interaction between the server and the client during the execution of the application. The front end component is sent to the client, which executes it to open up a TCP/IP connection between itself and the server. The server then starts to execute the application.

Thus, in the client-server environment of Gish, code is not generated in order to access code for executing on a server. Therefore, Gish does not teach, show or suggest ***generating, in response to identifying the code for executing on the one of the plurality of servers, code for accessing the code for executing on the one of the plurality of servers*** as in the claimed invention.

Further, in the client-server environment of Gish, a server is contacted by the client system to process the application. But Gish does not so much as suggest that the server contacted is one among a plurality of servers much less determining which ***one of a plurality of servers*** to contact to process the code. Therefore, Gish does not teach, show or suggest ***processing the generated code to determine the one of the plurality of servers on which to execute***

the code, each code for executing on a server being able to execute on a particular server as in the claimed invention.

Consequently, Gish does not teach the claimed invention, but for the first element, as asserted by the Examiner.

Therefore, it would not be a simple matter for one skilled in the art to incorporate the use of inputting object on a client system described by Gish to arrive at the claimed invention, especially when the client system of Gish does not determine ***one server among a plurality of servers*** to contact to process the code as in the claimed invention.

Therefore, Applicants submit that Gish does not teach, show or suggest independent Claim 1. Consequently Claim 1 and its dependent claims are allowable over Gish.

The other independent claims (i.e. independent Claims 12, 28, 29 and 32) and their dependent claims all include in one form or another the limitations of ***processing the generated code to determine the one of the plurality of servers on which to execute the code, each code for executing on a server being able to execute on a particular server*** and ***identifying, based on the accessed code for executing on the one of the plurality of servers, one of a plurality of client applications for allowing the determined server to interact with the client system during processing of the code for executing on the one of the plurality of servers***. Consequently, they too are patentable over Gish.

Nonetheless, Applicants would like to point out that the dependent claims are patentable of their own right. For example in rejecting Claim 2, the Examiner asserted that in col. 19, lines 13 – 19, Gish teaches ***using a view list of at least one input element for processing a type of code identified by the input object***. The Examiner further asserted that in col. 27, lines 51 – 55 and in col. 28, lines 20 – 31, Gish teaches ***processing the generated code includes using a server list of at least one server element for determining the server***. The Examiner then asserted that in Figs. 4 and 5 and in col. 18, lines 14 – 16,

Gish teaches ***identifying the client application includes using a launcher list of at least one client element for launching the one of the plurality of client applications***. Applicants respectfully disagree.

Again, Applicants would like to point out that since as admitted by the Examiner, Gish does not teach an input object, then Gish cannot teach **using a view list of at least one input element for *processing a type of code identified by the input object***.

Secondly, in col. 19, lines 13 – 19, Gish discloses:

> When application execution is initiated, the client node begins to interpret the PE 700 (FIG. 7) it has received from the server node 710. The PE 700 is a framework (which includes an User Interface (UI) which can include a graphical UI (GUI)) and an instance of a communication library 720 implemented in Java. Once it starts up, the PE 700 opens a socket connection to the server node 710 utilizing the server port number it was supplied when the server app manager 650 started the back end process 730.

But note that nowhere in the above-reproduced paragraph is there a reference to a view list with input elements.

Therefore, Applicants submit that Gish does not teach ***wherein processing the input object to identify the code for executing on the one of the plurality of servers includes using a view list of at least one input element for processing a type of code identified by the input object*** as asserted by the Examiner.

In col. 27, lines 51 – 55, Gish discloses:

> ICE-T applications can use a Java-enabled Web browser for client access to application execution. Although developers may choose to have applications launched outside a browser, a Web page presents a familiar and easy to use interface for launching applications.

Wherein in col. 28, lines 20 – 31, Gish discloses:

> When a user launches an ICE-T application, the client node establishes a Web connection with the server node using HTTP. The server manages this Web connection. ICE-T applications can be launched from a browser, an applet viewer, or as standalone applications. FIG. 26 illustrates the steps associated with launching an application URL in accordance with a preferred embodiment. On the server side, the ICE-T Access Layer (a cgi-bin executable) authenticates the user data. If the authentication succeeds, the Access Layer contacts the ICET Application Manager and the Application Manager starts the server program and initiates a network session.

In the paragraphs above, Gish discloses that an application can be launched inside or outside of a browser and when a client launches an application, a Web connection is established between the client and a server.

But nowhere in those paragraphs is there a teaching, showing or suggestion of using a server list that has at least one element to determine a server on which to process code.

Thus, Applicants submit that Gish does not teach, show or suggest ***processing the generated code includes using a server list of at least one server element for determining the server*** as asserted by the Examiner.

Finally, Figs. 4 and 5 are explained in col. 17, line 34 to col. 18, line 38. In the passages in col. 17, line 34 to col. 18, line 38, Gish discloses in general that a client sends a request to a server to execute an application. The server identifies, based on the application to be executed, a front end component and a back end component that will facilitate interaction between the server and the client during the execution of the application. The front end component is sent to the client, which executes it to open up a TCP/IP connection between itself and the server. The server then starts to execute the application. The server then starts to execute the application.

However, Gish does not teach a launcher list of at least one client element for launching client applications.

Therefore Gish does not teach, show or suggest ***identifying the client application includes using a launcher list of at least one client element for launching the one of the plurality of client applications*** as asserted by the Examiner.

Since none of the elements of Claim 2 are taught by Gish, incorporating the input object in the client-server system of Gish, as asserted by the Examiner, would not teach, show or suggest the invention as claimed in Claim 2.

Regarding Claims 3, 5, 14, 16 and 18, they all contain the limitations of one or all of a view list, a server list and a launcher list as being extensible. For example, Claim 3 includes the limitations "wherein at least one of the view list, server list and launcher list is extensible to accommodate additional respective elements."

The Examiner, in rejecting Claim 3, asserted that Gish teaches the limitations therein in col. 18, lines 14 – 16 and in col. 28, lines 20 – 26. Applicants respectfully disagree.

In col. 18, lines 13 – 16, Gish discloses:

> After authenticating the client 500, the server node 520 selects front 502 and back end 510 components based on the application definition list maintained at the server node 520.

And in col. 28, lines 20 – 26, Gish discloses:

> When a user launches an ICE-T application, the client node establishes a Web connection with the server node using HTTP. The server manages this Web connection. ICE-T applications can be launched from a browser, an applet viewer, or as standalone applications. FIG. 26 illustrates the steps associated with launching an application URL in accordance with a preferred embodiment.

In the above-reproduced two passages Gish does not disclose a view list, a server list or a launcher list, much less having a view list, a server list and a launcher list that is extensible to accommodate additional respective elements.

Thus, Applicants submit that Claim 3 is patentable over Gish.

Since Claims 5, 14, 16 and 18 each contain one of a view list, a server list and a launcher list as being extensible, they too are patentable over Gish.

In short, Applicants submit that the claims are patentable over the cited references and hence, once more respectfully request reconsideration, allowance and passage to issue of the claims in the Application.

Respectfully Submitted

By: _____
Volel Emile
Attorney for Applicants
Registration No. 39,969
(512) 306-7969